computing platform, which hardware, which software." At one point we realized that we had over half a dozen different BASICs in the corporation, none of them compatible, each aimed at particular markets by the division that sponsored it. We were horribly inefficient in attacking the market because a division could do anything it wanted; there was maximum independence and minimum coordination. The problems got worse as platforms and their divisions proliferated -- the hand-held division, the desk-top division. Our entry into the 32-bit computer finally came out of the desk-top division, because the Focus chip set was designed to power a computer that was ten times faster than the 9845, which was a single-user desk-top BASIC system, that we then tried to put UNIX on and make it a general-purpose "workstation" using a $100 million proprietary chipset -- instead of going with the then-standard 68000 from Motorola, and that's a whole different story....

THE MAC AND ME:
15 Years of Life with the Macintosh

---

by Jef Raskin

INTRODUCTION

The success of the Macintosh cannot be credited to any one person. I gave it its human-oriented, graphics-based, compact-sized nature from the very first, invented some now-universal interface concepts, and made many decisions that proved fundamental to its success. I hired a crew of unknowns who have become, almost without exception, men and women known throughout the industry for their continued innovation. It was not just me, but my original Macintosh crew of four, then a dozen or so, and finally hundreds of people, who created that first Macintosh. Now thousands at Apple continue to create and expand the Macintosh line of computers and the machines that will follow in its footprint. And even so it would have been a dead-end product, after all this effort, without the work done by thousands of software developers who give tens of millions of Macintosh users the tools they need. In this logarithmically spiraling cascade of numbers we come today to something over a hundred million people who use -- at their desks at home or at work, in their schools and libraries, at the beach, in airplanes, everywhere -- systems that look and feel much like Macs. Amplified by all this effort and the sincerest form of flattery, the influence of the Macintosh may well have touched the lives of over one percent of the world's population.

The phenomenon that I have just described represents the expansion of one person's stream of ideas into a flood, but the stream had to first gather force from numerous tributaries. It was not just my own inspiration, but the flowing together of the

genius of Ivan Sutherland and Douglas Englebart, the scientists at Xerox PARC, the development of the microprocessor, the success of the Apple II, the efforts of many other people whose work I studied and learned from (I will never be able to thank them all) -- and a lot of luck -- that led to that one nexus in space-time, in the spring of 1979, when I went to the CEO of Apple and told him that I wanted to design a new product I had been dreaming of for a while, and that I wanted to call it Macintosh.

In 1994 the 10th anniversary of the introduction of the Macintosh was celebrated with a rash of articles -- some of dubious accuracy -- and parties at Apple and elsewhere. But it was also the 15th anniversary of the origin of the Macintosh project. This is the story of how the Mac, a product that has changed the face and interface of computing, first came into being.

THE HUMAN-ORIENTED COMPUTER SCIENCE STUDENT

It's hard to say when the conceptual framework for the Mac began. Parts of it can be discerned as early as 1965 when I was a graduate student in computer science at Penn State. Already steeped in the technicalities of computer design and programming, I nonetheless found computers aggravatingly -- and unnecessarily -- difficult to use, and always looked for ways to make them less intimidating. I soon earned a reputation as being sympathetic and helpful to our least technical users, especially those in the arts and humanities. Since I was (and am) as comfortable in the humanities and the fine arts as I am with science and mathematics, I never forgot our shared pain and frustration with the nonsensical ways computers were (and are) operated. In contrast, most of my fellow students celebrated their detailed knowledge and seemed to enjoy the power and status that distinguished them from "ordinary users." They preferred to work with hard-nosed programming students with whom they could attack problems in full jargon.

In my 1967 thesis, "The Quick Draw Graphics System," I took issue with the display architecture then in vogue. At this time, input was mostly via punched cards, and output took the form of extravagant quantities of oversize paper sheets from massive and noisy "line printers." Those who wanted pictures turned to expensive plotters designed to do engineering drawings. There were only a few CRT terminals at the Penn State computer center, and these could display only letters and symbols, usually in green or white on a black background. Hamstrung by specialized electronics -- in particular a circuit called a "character generator" -- that permitted no other use, they could not display graphics. One display at the center could draw thin, spidery lines on its large screen. With it you could do drawings that now seem crude, annotated by child-like stick-figure lettering.

In this milieu my thesis was radical in suggesting that computer displays should be graphics- rather than character-based. I argued that, by considering characters as just a particular kind of graphics, we could produce whatever fonts we wished, and mix text and drawings with the same freedom as on the drawn or printed page. To prove my point, I wrote a program that generated the complex, two-dimensional notation of music. To accomplish this, I needed to enter graphic data into the computer system.

Commercial digitizers were then as expensive as a small house; my only choice was to design and build one. Its input was somewhat indirect, in that as I pointed here and there, it produced punched cards which had to be read into the computer later. With only limited access to a machine shop and within the tiny budget a graduate student might worm from the university, I found it hard to achieve the required precision and repeatability; but my digitizer, although mechanically and electronically Rube-Goldbergish, was an inexpensive and practical one-point-at-a-time Graphic Input Device (GID). I did not know of [Douglas] Engle-bart, on the West Coast, and his recent invention of the mouse; even if I had, it would have been hard to hook it up to the mainframe we were using.

A mark of how much things have changed was my casual use of the word "fonts" in the paragraph above. Today, almost every computer user thinks of character display and printing in terms of fonts; but when I was a graduate student -- and even when I started the Macintosh project -- most people in the computer world did not think of fonts in connection with computers. When I talked about the merits of serif and sans-serif fonts, the advantages of variable- over fixed-pitch fonts, or the beauties of Bodoni's work, I got blank stares and people might mutter. "There goes Raskin with his odd art stuff again." To talk about fonts and drawing was to emigrate from computer science to the world of graphic artists, typographers and other "arts people." Now, everybody seems to have a few dozen fonts on their computer to play with; what I wished for has happened.

Another radical claim I made in my thesis was that ease of use should have a higher priority in the design of computers than speed and efficiency. Learning how to make code run in shorter time, or less memory, or both, was central to computer science training; human interface was not given the slightest consideration. Computer time was expensive then, and pride of place for human convenience was an alien concept. It was not unimportant at that time to use internal computer resources efficiently, and it is still essential today. But efficiency should be neither an end in itself nor the highest ambition of the computer scientist -- contrary to the impression one often got in graduate school in computer science.

Old-fashioned computer centers were an ideal breeding ground for pranks. Appalled by the daily waste of paper, a few friends and I

once decorated the computer center building with a day's discarded output. Early arrivers the next morning found a band of white interrupting the red brick, and had to do some tearing to get into the building. The same stunt now could be considered a work of art. On another occasion a state-level dignitary was visiting the computer center. I set up the computer so that opening the massive printer cover (done by a remote command) would dump a wastebasket full of the punched-out paper chips from a card-punch into an air vent intake, giving the startling effect of a short-lived snow storm coming up from the floor. No one doubted the identity of the perpetrator (I guess I had a reputation) so sweeping and vacuuming were my lot.

My friend and office-mate Steve Zins and I engaged in a rubber-band gun arms race: my best designs were single-shot guns of unprecedented accuracy, needed to shoot the flies generated by the adjacent cow fields. Steve created a Gatling gun that could plaster my chest with some 60 rubber bands in less than a second (though the gun took five minutes to load for that one burst.) But I digress...

TO THE WEST COAST

The first truly interactive graphical computer system that came to my attention was Ivan Sutherland's Sketchpad. Though the hardware available at Penn State would not allow me to follow his lead, Sutherland's work was a revelation and an inspiration. It used a CRT display and had a light-sensitive "pen" for graphic input. In high school I had built a rudimentary light pen for an oscilloscope, so I immediately knew how it worked. One peculiarity is that you had to put up a mark of some sort (Sutherland used the word "INK") so the pen had some light to detect. You just could not start drawing without first pointing to the "ink," after which the computer could track the light pen. If you tilted the pen too far or moved it away from the screen, the computer "lost" the pen.

These details must be emphasized. Without them it is too easy to imagine, when you hear that Sutherland's ground-breaking system had "rubber band" lines and could do graphic input and output, that it all worked in the now-familiar Macintosh and Windows fashion. By present lights Sutherland's system was crude and limited. In its own day it was a wonder and an inspiration.

As I fretted with the details of getting my thesis approved, I began dreaming of a computer that would be graphical, easy to learn, easy to use, capable of everyday tasks such as word processing, and, above all, affordable. At the time this was not just a dream, but simply impossible. For a while, getting my degree also appeared an impossible dream; my thesis was rejected for not following the rules. One rule in particular was that you were to use only one font in a thesis; they didn't want you to

turn out part of it on one typewriter and the rest on another. This had launched a local industry of typists who knew the university rules to the letter, and whom you paid to produce the final draft exactly to specifications, in the required number of copies.

My thesis had characters in several fonts, exactly to demonstrate that one could produce distinct fonts on a graphics-based system. The use of varied fonts was, I complained, part of the subject matter; to rule out their use was to attack the content, and not just the form, of the thesis. After months of verbal wrangling and a memo war, my thesis was accepted, fonts and all.

Tired of Pennsylvania winters and Penn State's cold bureaucrats, my wife and I drove west, until we ran out of land in La Jolla, just north of San Diego. We knew nobody in the area, but by luck had ended up at the University of California's Scripps Institution of Oceanography. Walking out onto the Scripps pier I saw, for the first time, a pelican abruptly fold its wings and splash into the ocean. I thought it had been shot.

EARLIER INFLUENCES

Sometime in middle or early high school I was given a copy of Claude Shannon's marvelous _Information Theory_. I can remember no other books from that time, by both title and author, except Arthur Conan Doyle's Sherlock Holmes mysteries. It was eye-opening and completely wonderful to learn that this ephemeral, seemingly unquantitative stuff called information was amenable to a physics as rigorous as that for objects and motion -- and that this physics was almost purely mathematical in its development. This was extremely appealing, as mathematics was by far my first love, queen of all the subjects I could command. It may seem premature for one at the age of 14 or so to be so smitten; but it was my good fortune that Ron Genise -- a most wonderful teacher, and later friend -- had begun, in my sixth grade, to make the beauty and power of mathematics as alive and vivid for me as the performance of sports figures and cars were for my classmates. I believe that he first pointed out the Shannon book to me, and if my memory is astray on that point, at least I know that he led me to the intellectual point of view from which I could appreciate it.

During this time I read an article about the rate at which information (measured, as Shannon had shown, in bits per second) could be communicated from the eyes to the brain. The number seemed much too low. For example, I could sight-read pieces by Chopin and Beethoven on the piano. As an early exercise in information theory I calculated the number of bits in each symbol. This is not difficult; for example, a note head specifies one of the 88 notes on the piano, and this takes a little over seven bits. It also specifies a duration, which for most

practical purposes has one of 8 values, which is exactly three bits of information. So a note conveys approximately 10 bits of information.

Ignoring other symbols and some details here (I was more precise in the paper I wrote at the time) and considering music where one is reading four chords each of six notes in a second, implies a transmission rate of 2400 bits per second (2400 baud.) This exceeded the rate at which neurophysiologists believed the senses could transmit data. This made me realize that I wasn't really reading each note, but analyzing the chords into harmonies: "that's an E-flat major chord in the first inversion..." and so on. All my brain had to deal with was the single concept of a certain chord, and not all the details of each note. Later, reading about psychology, I found that I had re-discovered a phenomenon called "chunking" which allows us to grasp much more than would be indicated the slow data rates experiment shows our brains can handle.

This has been a paradigm typical of my entire life; supposedly different disciplines merge or interact, reinforcing each other. Studying math opens the path to a book on the physics of information which informs my classical musical studies, enhanced by my having ignored my music teacher's wishes and learned to play from jazz "charts" instead of sticking only to classical music. The speed at which I can read music seems to violate a fact I read in a science magazine (I am a compulsive reader, and will read almost anything,) the solution to which gives me an anchor of understanding when I am learning about something in psychology years later. Somehow it all fits together.

FAMILY AND FEMINISM

My parents brought my brother and me up to recognize oppression and to fight it. Popular, gregarious, and very active in civic events, they risked friendship and fortune in defending racial equality in the 1950s and 60s.

The following is part of the column I wrote about my father for the local paper:

My father, Bill, died last week. It was not at all unexpected. His health had been failing since my mother died a few years ago. Recently he had had a stroke and was also diagnosed with congestive heart failure. He could barely speak. My brother Michael had flown in from Boston and the three of us were together for what was to be the last time. Bill struggled for speech and repeated, "What can I say? What can I say?", a phrase he had always used when overcome with emotion. We told him that he didn't have to say anything, but he finally said, with evident

effort, "I love you." and embarrassed us a little by taking our hands and kissing them All I could think to do was to return the gesture and kiss his hand. At this he smiled his delightful smile, made lop-sided by his stroke, yet a smile that reminded us for a moment of the father he had been.

If you had met him you'd have found a mild-mannered man, soft-spoken, well-liked and without self-interested ambition. The love between my parents was constant and evident to all who knew them. A responsible citizen, a merchant, a member of the school board after my brother and I had gone on to college -- he would not be on the board while we were students to avoid tainting our achievements with suspicions of favoritism. For many years he was the secretary of the Lions Club. He never accepted the many nominations to be president. He liked to lead by example, by quiet persuasion, and with gentle humor from the sidelines.

On moral issues he was inflexible; I have space for only one example. In the '50's, long before the present civil rights movement was in full steam, he incurred the enmity of nearly the entire town by supporting the hiring of man of African descent as an English teacher, and backing another as a member of the Lions Club. I remember a long-time customer coming in to our store and saying, with true regret in her tone, "I can no longer shop here. You understand why."

We had a solemn family meeting. Our parents told us that we had a choice to make: if we continued to back our beliefs we would be very poor for a while, there would be no toys at year's end, no going to restaurants, and so forth. We knew what he meant, our family-owned store was too often quiet, the piles of layaways for Christmas were not building up in the basement as they had in previous years. The other choice, he said, was to hold onto our beliefs privately but not push matters. He would not impose his values and the attendant risks on his children. It was up to us and we knew he would abide by our decision. Michael and I had no patience with people who judged others on their race or cultural background, and we said (as Bill proudly recounted years later) without hesitation that we didn't care about presents but that we did care about our friends. To do nothing was to give tacit approval to racism.

Some would say that we lost. We had to sell the store across from the railroad station and set up shop in a poorer neighborhood. Instead of big Buicks and Packards we drove the cheapest Renault. We no longer had a summer house by the lake. And the lovely presents we had become used to came no more. The Lions club split in two, a large whites-only club and a tiny integrated one with my father as secretary. But we won. The high school had its first black teacher, and others followed. The white Lions club faded and Bill's survived.

There were no services; he was an atheist as righteous as any church-goer. He donated his body to science, and his love of humanity to his sons.

The messages were clear. One was: figure out what is right and then stick to your guns. Another: principle is more important than practicality. These were two of the beliefs that propelled the Macintosh as it came into being. I was not uncomfortable defying common wisdom.

This early training also made it easy for me to recognize girls and women as an oppressed class in our society. As a child I had seen my intellectually brilliant cousin, Miriam, given dolls while I would get the far more interesting Erector sets and chemistry labs we both preferred. My feminist leanings were deepened by some of the things that later happened to my friend Karen Kalinsky. For example, when we were undergraduates at S. U. N. Y. at Stony Brook, I held a job in the computer center. Karen, also a math major, became interested in computers and decided to take the programming course; the head of the computer center there had offered jobs to the people who got the three highest scores on the final and Karen was rarely outscored on any test. Looking at the posted list, we saw that she had received the top score, and we anticipated working together at the computer center. The jobs, however, went to the three top men in the class! We were mad, she raised a ruckus, and I quit in protest.

We went to Penn State next. As usual, I got a job at the computer center, and as usual, they wouldn't hire her -- in spite of credentials better than mine in some ways, such as grade point average -- because her "boy friend" worked there. After we were married, there was no way she could be hired. Nepotism, you know.

By the time we reached the west coast, degrees in hand, we were smarter about jobs. She took a position first, running the computer at the Institute for Geophysics and Planetary Physics at University of California at San Diego (UCSD.) Shortly thereafter, I got a job at the University Computer Center. They didn't think to ask a man if his wife worked in a professional capacity -- on the form it only asked if your husband worked at the University. I also saw some of the sexist hurdles my cousin Miriam had to face, such as being ignored by the professors in classes. (Miriam is now a professor and researcher at the University of Michigan at Ann Arbor.) These experiences are relevant to our story; for one thing, they influenced my choice of the name "Macintosh" for my favorite computer.

While working at the UCSD computer center, I became familiar with other parts of the school. The music department, filled with avant garde composers and performers, was intrigued by my back-

ground in both computers and music (I had designed and built the first electronic music studio at Penn State.) It looked like a good fit and I became a graduate student there, working toward a Ph.D. in music. UCSD, like some English universities, is divided into colleges; at the time, the first two were named Revelle and Muir Colleges, and the newest was simply called "Third College." By a peculiar series of events (that would, for once, take us too far afield if I described it here,) I soon became the computer center director and a professor of Visual Art at Third College, positions I held from 1969 through 1974.

The main computer center at Revelle College was noisy, antiseptic, and lit by fluorescent lights that glared off white vinyl floors. It was punch-card-oriented and built around a physically huge, multi-million dollar mainframe computer. The computer center I designed for Third College, located in a war-surplus Quonset hut, was very different. It used a pair of Data General Nova minicomputers with 16 interactive terminals. The decor was bean-bag chairs and Japanese paper lanterns, giving my center a friendly, funky feel. It became the natural home for people with what were then seen as "odd" computer applications, like music and art. Some of the campus's computer aficionados found that they preferred the unhurried, interactive context of the minicomputers to the fluorescent, buzzy mainframe environment on the other side of campus. In light of today's personal computers, which operate in homes, cars, and at the beach, it is hard to remember that in the early 1970's a computer center such as the one I created was countercultural, and perhaps unique.

THE THIRD COLLEGE COMPUTER CENTER

My computer center was funded primarily by the National Science Foundation and the University of California. I suspect that if Senators Proxmire or Helms had ever visited it they would have mistaken it for a typical waste of taxpayer's money. It looked more like a place to get stoned than to get educated, a hippy haven.

But looks are just looks, and I have always made my courses friendly in spirit while I demanded hard work from the students. The new computer center was an effective educational facility. Only ten per cent of the undergraduates who learned programming at UCSD went through my courses, but nearly half of the students who held paying jobs at the main computer center had done so. There was no doubt that the Third College computer center was doing a good job at creating future computer scientists. Better still, it was reaching students who would otherwise never have gone near a mainframe, just as the Macintosh would someday be used by people who thought they'd never touch a computer. It was not the technology that made my teaching so effective, but the interface! Students learned more and better in a pleasant environment where, to test their programs, they simply pressed a

key and got results. The other side of campus was batch-oriented; you presented a deck and went to the printer to await your output. Fortunately Ken Bowles, the director of the Revelle center, had an enlightened attitude for someone in his position at the time; he did not regard a second, student-oriented computer center as a challenge to his hegemony. Years later Bowles would spearhead development of the computer language and operating system called UCSD Pascal, which would be essential to the success of Apple and the Macintosh.

In retrospect, the Third College Computer Center was all that a grant administrator could wish for: it met its educational aims, resulted in appropriate publications, and later went on to inspire commercial products that have boosted the GNP (gross national product) to the tune of billions of dollars. It is definitely possible to see precursors of the Macintosh in the Third College center's low tables with small rectangular monitors and detached keyboards. Though they had to be tied to a common system, the effect was as if each student had a personal computer. Resources had to be shared in 1973, when a 4K byte (enough to hold about 800 words of English) random access memory (RAM) memory unit cost nearly two thousand dollars. As this is written, each 4K bytes of RAM in my Macintosh computer costs less than 10 cents.

During the summers, I used one of the Novas as my personal computer. My mostly volunteer staff and student friends (notably Jon Collins, Barbara Zakarian, Bill Atkinson, and Steve Clark) helped me put the computer into the back of my truck on a wheeled dolly, and we used it wherever we went. One memorable time we took it with us into a restaurant, using it to figure the bill and the tip, to the amazement of the waitresses and patrons who crowded around. A computer outside of a lab was an absolute novelty. These experiences with a "portable" computer system gave me a foretaste of what it would be like to own a personal computer. Like the crocodile in Peter Pan, I would never forget that taste, and craved it for years.

SAIL AND SILICON VALLEY

In 1972 I visited the Stanford University Artificial Intelligence Laboratory (SAIL,) which had an established reputation as a center for advanced research in computer science. I was also introduced to another magical place that had recently opened and was a short bicycle ride away. The Xerox Palo Alto Research Center (PARC) was to become even better known than SAIL in the coming years. Thanks to a strong common interest in early music as well as computers, I soon found a close friend in the person of Doug Wyatt, a tall, thin man who is as quiet as he is technically brilliant and musically talented. Doug took a leave of absence from PARC and came down to San Diego for a while to write new software for my computer center.

A programming language I designed, "FLOW," was implemented and improved by Doug. The human interface used in this system, as well as the design of the language itself, were somewhat ahead of their time. It proved so effective that it came to the attention of the cognitive psychologist Don Norman, later to become a leader in the fields of cognitive psychology and man-machine interfaces, who is now an Apple Fellow and a writer of popular books on the subject. Norman did some of his first computer-interface-related work investigating why students learned faster and better with the "FLOW" computer language.

The next summer I was invited to become a Visiting Scholar at SAIL. It was a great place to be. I remember fondly the memorable daily, end-of-the-day volleyball game, after which most of us would retire to the lounge and watch _Star Trek_. After which a lot of us would get supper and go back to work. There are a number of reasons why I remember watching _Star Trek_. I enjoyed the show, and once it led to a remarkable incident.

To understand what happened, you have to know that an experimental robot occasionally roamed the halls and parking lots at SAIL. The rambling robot (in case you were picturing C3PO walking across the desert with R2D2) looked like a wheeled table full of surplus electronics. It was not at all humanoid, or even robotoid. On this occasion we were sitting down to watch Captain Kirk and his enterprising crew when the robot wandered in, stopped, swiveled its TV eye at the set and sat there throughout the show. At the end, it whirred into life, rolled itself around, and left as we did. Later I learned that Hans Moravec was working at a terminal that did not have TV feed (most terminals at the AI lab did -- another development way ahead of its time that I was exposed to) and had sent in the robot to beam the picture and sound back to his monitor. It occurred to me that we were probably the only people in the universe watching _Star Trek_ in the company of a robot.

While at SAIL I used the early Defense Department progenitor of the now-popular Internet. ARPAnet allowed us to communicate with and use remote computers. It felt like magic to be sitting in California and running a computer at MIT. I became an early e-mail junkie, a habit that I have yet to kick after 20 years.

PARC

The populations at SAIL and PARC intermingled freely and I found myself gravitating more and more toward the beanbag chairs at PARC. A significant portion of their work was based on the same goals as my own, to make the power of computers accessible to non-specialists. I suspect that I fit in easily and well because they didn't have to start by converting me to their point of view; I was already there. I meanwhile felt at home since, for

the first time, I was among computer scientists who were on the same wavelength as I. They had accomplished independently what my thesis had called for a few years earlier: computers that were graphic-based, without impediments such as character generators. What was more exciting was that the people I spoke with were concentrating on interface design, which I also saw as the area of computer science most in need of development.

By 1974 I was fed up with the politics in the UCSD art department and left the University, making my point in artistic fashion by ascending in a huge hot air balloon, playing the sopranino recorder, and announcing my resignation from 100 feet up in the air. I was also tired of the directions the computer industry was taking. It was all "more" and "bigger" and "faster," but not really better (this history is being repeated with personal computers today.) Nobody seemed interested in what I was preaching about usability. I sold my house near San Diego and moved to Brisbane, a town just south of San Francisco. I tried the life of a street musician and music teacher, started a company that made radio-controlled model airplane kits (a business that continues today,) and became the conductor of the San Francisco Chamber Opera. I also worked briefly as a packaging designer, but left when I couldn't convince the owner that we could design boxes faster and better with a computer. In the 1990's the owner's son, who better understood what I had proposed, wrote a set of computer programs to do box layout, and now has a successful business making boxes -- and an even more successful one selling the software.

I also worked as an advertising and portfolio photographer (having been taught a bit about the art by my former student and forever mentor David Wing, now a professor of art at Grossmont College east of San Diego.) During this period of wandering, I started a company called Bannister & Crun to write software and manuals. The company was named after two characters (Minnie Bannister and Henry Crun) featured on the BBC's beloved Goon Show. Between the way the Goon Show's players mangled English and the spotty reception of my shortwave radio, it was sometimes hard following their humor; they were to radio what Monty Python was to become to TV.

Our first job at Bannister & Crun was to computerize the South San Francisco sewer billing system, a job that required me to visit the sewage treatment plant from time to time and work on one of the most dreadfully designed computers I had ever seen, an early Qantel model. We also worked on other software projects and wrote manuals for companies that included National Semiconductor and Heathkit.

ENTER THE MICROCOMPUTER

In late 1974 the general purpose microprocessor chip was put on

the market and I remember discussing its incredible potential with Doug Wyatt and a mutual musical friend, a talented and extraordinarily pleasant man named Brian Howard. Of broad learning, with a degree in Electrical Engineering from Stanford, he was working for the preventive medicine department at the university, doing a characteristically wide range of things including building test equipment. Brian was to become a central intelligence in the development of the Macintosh and, later, one of the designers of Apple's first laser printer -- another product that changed the face of computing. He continues as a respected engineer at Apple.

When the first microcomputer kit, the MITS Altair, was announced in 1975, Brian, Doug, and I just had to have one. With soldering iron, oscilloscope, and logic probe in hand, Doug and I built the Altair and (somewhat to our surprise) got it working. This was a non-trivial endeavor, but Doug's combination of methodical care and clever insight solved many a problem. I was experienced with a soldering iron and felt comfortable with the construction because electronics had been a hobby of mine as a child. I had won a science fair prize for a computer I made while in high school. Building a computer is, perhaps, nothing to crow about now, but in 1960 individuals just didn't have computers and kids didn't use or program them. I was still a hardware jock as an undergraduate, designing and building a computer from scratch for the Biology department at the State University of New York, then at Oyster Bay (now Stony Brook.) This background proved useful when creating the Mac, since I had a realistic idea of what could and could not be done with electronic components. Having done electronic design and testing myself, I could communicate with electronic wizards, and not be snowed when they spoke of impedance or logic levels.

We got the Altair running a program that did stock market analysis, and we sold it for about $5,000 to Jim Hurst, a stock market guru. He'd been paying $10,000 per month in time-shared computer charges for the same work, so the micro system amortized out in two weeks -- a great savings to him. The computer had cost us a few hundred dollars and had served well as an introduction to microcomputing. With part of our profits we bought a slightly more sophisticated IMSAI, and I built the first modem kit that became available. I made back a bit of the money I spent on that kit by authoring a review of it for _Dr. Dobb's Journal_. I liked reviewing kits and I was soon writing the "Consumer Notes" column for that magazine. I became a reporter on the early personal computer scene; pieces I wrote appeared in _Personal Computing, Interface Age,_ the _Silicon Gulch Gazette, Kilobaud, Datamation,_ and _Byte_ magazine.

Jim Warren, who ran the wonderfully-named _Dr. Dobb's Journal of Computer Calisthenics and Orthodontia_ (Running Light without Overbyte,) is a delightful, jovial, and unconventional man who

sparked much in the industry. He created the West Coast Computer Faires and now works on creating political enfranchisement through technology. He often managed the Faires by cruising their huge exhibit halls on roller skates. One of the assignments he gave me in 1976 was to interview two fellow-members of the now-legendary Homebrew Computer Club centered in Palo Alto. The club, many of whose members went on to become prominent in the computer industry, was moderated by the very funny and genial Lee Felsenstein. This was also the man who designed the modem I reviewed (it's a small valley.) Doug Wyatt and I got an ovation one night when I announced that we had run our IMSAI for over a month without once taking the top off to fix something; it was a real milestone (and a tribute to clean soldering.)

THE TWO STEVES

The members I was sent to interview were building a new computer in their garage. By coincidence both were named "Steve" and their project was the Apple I. I was impressed by Steve "Woz" Wozniak's brilliant and efficient design and pre-decoded bus concept, and his exposition of the advantages of the 6800 and 6502 architecture over the competing 8008 and 8080-based machines. (Incredibly, this competition between architectures continues to this day.) I remember Woz explaining how the pre-decoded bus made peripherals simpler, that you could send information to peripherals the same way you wrote to memory, and that memory wasn't paged -- unimportant details in this essay perhaps, but indicative of the kinds of considerations that Woz paid careful attention to. And I loved the name "Apple" instead of the techie names everybody else was using; it fit my kind of iconoclastic spirit. Now we have become so accustomed to it that it is hard to remember how joltingly countercultural that name was at first. A computer company named "Apple"?

The other Steve, Steve Jobs, was a delight to talk to about less technical aspects of computers. His enthusiasm and business orientation were exciting. They were just starting on the design of the Apple II, and I tried to convince them that they should employ bit-mapped graphics and not have a character generator, but Woz thought that software couldn't handle the character generation task fast enough and Steve Jobs didn't understand why I thought it so important. I had a different vision of what a microcomputer should be like, and PARC's programmers and my own work had convinced me that software could do the job. I tried to convince Woz by working out the code to put bit-mapped characters on the screen and calculating timings by counting cycles, but the Steves were not open to the idea. The concepts I espoused were far from the mainstream of computer design and for all their mold-breaking thinking, Steve and Steve were very strongly conditioned by the minicomputers they had seen. To do them justice, Woz was absolutely correct in stating that a character generator was much faster and its software less memory intensive

than my all-graphics approach. But had I been able to convey my vision better, I suspect he could have made bit-mapping work fast enough back then.

It was by the slimmest of chances that the Apple II had a high resolution graphics mode (Hi-Res) on which bit-mapped graphics could later be explored. Woz was not going to include it but Jobs asked Woz how many chips it would take to add the feature. Woz said that it would take only two, so Jobs insisted that they could afford it. Sometimes history stumbles along from accident to accident, things are done that seem like a good idea at the time, and every now and then they are.

I tried to convince Jobs and Woz to visit PARC, which was a very academic and open place (Xerox may later have felt that PARC was too open,) but did not succeed. Jobs repeatedly told me (and anybody else he could get hold of) that a large corporation like Xerox couldn't do anything interesting. Hewlett-Packard's rejection of Woz's proposal for a personal computer, when he worked there, was a prime example of such corporate blindness, and ever after remained part of their psychological motivation. If I could have told them then that Hewlett Packard would someday make millions of dollars simply by selling peripherals to Apple computer products (as has happened,) the Steves would have been ecstatic, and rightfully so.

APPLE MANUALS

I worked with Jobs and Woz and, under the aegis of Bannister & Crun, wrote a user-oriented portion of the manual for the Apple I. There was a tiny misunderstanding about the price: I was talking about $50 per finished page and they thought I had said that it would cost $50 for me to write the whole manual. We resolved our differences amicably and work proceeded.

At about this time Jobs made a decision crucial to the history of personal computers. Paul Terrell ran the Byte Shop in Mountain View, one of the first retail computer stores in the world. When Jobs and Woz asked his advice he insisted that the Apple II must have a non-rectilinear, consumer-oriented, plastic case and -- unlike the Apple I and many of its competitors -- should never be sold as a kit for which potential users had to scrounge parts at local surplus or electronics stores. I well remember the hassle of finding keyboards that worked properly with the very early microcomputers, and connectors to fit the idiosyncratic circuit boards. The Apple II, Terrell suggested, should circumvent these problems by being factory-built with an integral keyboard and power supply.

Terrell was not quite alone in recognizing these desiderata. The SOL, designed by Lee Felsenstein and manufactured by Processor Technology, had a typewriter look. A Utah company, Sphere, sold a

complete little machine with a programmer's hexadecimal keyboard (base 16 numbers only) and included a video screen even before the Apple I was released. The Commodore PET and TRS-80, both designed with much attention to consumer needs, followed soon after the Apple II. But though Apple was not alone, it had important advantages. One was Woz's remarkable BASIC interpreter with color graphics commands embedded in it. Another was that Apple was led by a raving firebrand in the person of Steve Jobs -- which was just what the industry needed.

Bannister & Crun was engaged to write the manual for Apple II's BASIC. This gave me the chance to put some of what I had learned as a computer science professor into a vehicle that I believed would reach tens or hundreds of thousands of people in a few years. It had been hard to give up teaching, which I love and yet hope to get back to, but (as I wrote to my parents) I felt that I could do more good for education by working at Apple than in any other way open to me.

The BASIC manual, first published in 1978, turned out to be a trend-setter. Instead of starting off with the then-customary explanation of the internal architecture of the computer, it got right to what people had to do to get the product working. It first explained in a step-by-step manner how to hook up the computer and use the keyboard. It then quickly moved the learner into doing color graphics (in 1978!). Another first: the manual used color illustrations and photos.

Today, when computer products are graded by magazines on the quality of their documentation, it may be surprising to learn that the nascent company barely saw the need for an Apple II manual at all. Mike Scott (Scotty,) a large man whose occasionally high-handed manner and gruff speaking style could be intimidating, had come from National Semiconductor to be Apple's president. He said, half seriously, that at National they had done very well with one-page data sheets, and I could save the company a lot of money by doing likewise. I soon learned that in spite of his manner, he was open to cogent arguments. Later he was to protect and nurture my Macintosh project when it was at a delicate stage.

Writing user documentation was a perfect prelude to creating the Macintosh at Apple. Doing manuals forced me to look at each product -- in excruciating detail -- from the customer's point of view. It is an experience I wish all computer and interface designers could share. Any design flaw that interferes with learning or using the product becomes painfully apparent as you struggle to explain the quirk to the user. Time after time Brian Howard and I would wrestle with these problems, our frustrations coming out as subtle, snide remarks about design errors -- remarks that, in those innocent days, often appeared in our manuals. This sometimes annoyed marketing people, but it actually

served the purposes of Apple's products. The comments told the truth, showed sympathy for the customer's plight, and created credibility for the rest of the manual and the company as a whole. Too many manuals are fairy tales about how a product is supposed to work, or how it worked in the previous version.

When we wrote the Apple II manuals at Bannister & Crun the product was already finished -- we weren't trying to write a manual from specifications or rough prototypes. Trying to document a product still in development is an often-made mistake which guarantees a second-rate result. Since almost everybody now does this, customers have come to accept such manuals as standard. It can't work well, since you are documenting something different than what the customer will get, you are writing about a fiction, a planned product (and we all know that products always turn out exactly as planned.) To be sure, the manual can be edited to conform with changes, but that is not nearly as good as having the whole picture in mind from the beginning. Besides, you never catch all the changes, as the customers eventually find out.

We weren't the only group writing good manuals; another example was the superb HP 35 manual. The HP 35 was the first scientific pocket calculator, another fabulous product that opened up an industry. Its manual was an inspiration in terms of writing, use of color and layout, and informal conversational style. Instead of a lecture about the calculator's remarkable stack architecture or revolutionary custom electronic chips, the manual started you out punching buttons and seeing what happened. Later, when the topic had some value and experiential basis, you were given a mental model of what was going on inside. Along with my Apple I and Apple II (serial number 2) I keep my HP 35, still in working condition, in my office. The inspiring manual is displayed alongside it.

The Apple II manuals also worked because they were tested with typical users and rewritten as necessary, a concept nearly unique at the time in the computer industry, and one now regularly abandoned (to the detriment of users everywhere) under the excuse of time pressures. The time thus "saved" is not always a win; what the manufacturer gains by a few weeks' shorter product cycle is lost doubly -- to customer dissatisfaction, and as a continual drain on the bottom line attributable to increased support costs. As CEO of Bannister & Crun I demanded that I have a real product in its packaging before I wrote a manual, and in those days I got what I asked for. Errors in the manuals were extraordinarily rare thanks to these procedures. Brian Howard turned out to be a great editor, along with his other talents; his comments and those of Doug Wyatt were my education in how to write clearly and simply. My writing has never achieved the standards they set, but inasmuch as it is better than it was, they -- and the many other people who have since bravely waded through my first drafts and

let me know in no uncertain terms that a lot more work was required -- deserve a lot of credit.

APPLE'S MANAGEMENT

Having already approved the use of high-quality, coated paper, four-color illustrations, two-color printing throughout, and full-length manuals, management was reluctant to support the use of a wire binding so that the manual would lie flat (at least we had gone beyond flat lies.) I had often observed that most users didn't have a third hand to hold the manual open as they typed. During 1977, when I was merely a vendor to the company, two key people supported my point of view: co-founder Steve Jobs and chairman "Mike" Markkula, also known by his initials "ACM." Markkula had profited significantly from his experience at Intel, not only financially, but as a well-polished manager. Not having been in industry, I had never worked with a person of his extraordinary business skills, and I am still striving to live up to some of the examples he set. For one, he always gave the impression of having all the time in the world to hear what I had to say. He proved that he was listening, either by acting on my suggestions, or by taking the time to explain to me why they were not good ideas. One of my not-very-good ideas was to lower the price of the Apple II. I had been upset when I figured out how large Apple's margins were. Markkula patiently explained that while Apple's products were more expensive, the resultant financial strength of the company meant that Apple would be there for its customers in the future. It would have the money to develop new products and successfully market them while its competitors, who seemed to be doing a favor to their customers in the short term, would soon be out of business. He was right.

Jobs, in the early days of Apple, was an adamant protector of my writer's prerogatives who championed the need for testing and revision when Scotty didn't see things my way. Thus protected, I did the manuals as I thought they should be done, and Apple got what the press -- and even other companies -- praised as the best manuals in the business.

DEPARTMENT BUILDING

In mid-1977 I was still running Bannister & Crun, but my writing for Apple and the magazines had made me pretty well known in the industry, and I had lots of job offers. Chuck Peddle, leader of the PET computer project at Commodore, wanted me for a position there. Steve Jobs, who is as persistent a person as I've ever met, kept on asking me to join Apple as head of their publications department. I repeatedly declined, and he eventually asked what it would take to get me to join Apple. To put him off I made an impossible list which included an office with a window and a musical instrument, time to play gigs (I didn't want to let my musician friends down,) flexible hours, Apple's hiring

everybody at Bannister & Crun who wanted a job at Apple, and so on. He simply agreed to all my conditions, which I then wrote down; as it turned out, I should have done this with more of his promises. Bannister & Crun became Apple's publications department with me at its helm. I joined on the 3rd of January, 1978 as Apple's 31st employee. I presented no resume and signed no forms. Apple did no checking on my background. That I had led a team that had produced the nascent industry's best manuals was enough.

One day I heard that a new product, called the Apple II Pro, was being put together in the lab. From some technical details, I surmised that it could not possibly work as expected. So I snuck into the laboratory and turned on the prototype. Sure enough, it didn't do what it was supposed to. I went to Mike Markkula and told him that the machine wasn't up to snuff, and he replied that I couldn't be right, his engineers had assured him that all the problems had been solved. He was actually making plans for marketing and shipping the product and was about to start taking orders from dealers.

I took him to the lab and demonstrated my discovery. Upon talking to the people working on the project I had discovered a classic management nightmare (though it was new to me at the time): the engineers working on the project said that while the project was mostly going OK, there were still some unresolved problems. The next level reported to their bosses that a handful of problems would no doubt be rapidly fixed; they in turn told Scotty that it was nearly done, and Scotty told Markkula that it was just about ready to roll. In a more mature company I would probably have been fired immediately for my end-run around the hierarchy, but this time I was able to make a case for a "New Product Review" department. This would do for systems and software what QA (Quality Assurance) programs did for circuit boards and mechanical assembly. Suddenly, I was managing two departments.

Computers are not terribly useful without software (my definition of a computer is "a box for running software".) I argued that Apple would need to provide something new, application software, if we were to sell computers more widely. I created what may have been the first application software department at any microcomputer company. I tried to convince Apple to buy Visicalc, the first spreadsheet, when it was offered to us, but was out-gunned by Jobs and Markkula. It was Markkula's theory -- at least as he expressed it years later -- that to become a major application provider would have put a damper on third-party software developers, in the long run hurting Apple. What he said at the time I do not remember, but I do remember remaining unconvinced. With Markkula's approval I took a brief leave from Apple, arguing that if I could help make Visicalc a winner, Visicalc would sell a lot of Apple II's. As a result, I got to write the tutorial portion of the Visicalc manual, reporting to Dan Fylstra. Visicalc did sell a lot of our computers,

established a new category of software, and -- since it was a business application -- greatly helped the credibility of microcomputers in general.

In 1979 I found managers for two of my departments and became manager of Applications Software. Meanwhile, I was chafing at the limitations of the Apple II. The publications department managed to keep a secret that would have been embarrassing to the company had it been revealed at the time: the publications department was using not Apple IIs but Poly 88 computers. We were running a word processor I had designed and which had been implemented at Bannister & Crun. The Polys were a competing microcomputer that could handle both upper and lower-case letters -- a necessity in manuals. Due to mediocre design (they had no Woz,) poor marketing, and less imaginative management, they were soon out of business. Back in the garage days, in 1976, I had argued that the Apple II must have lower-case letters, but Woz disagreed. I held that the single biggest use of microcomputers would be word processing, he claimed that they would be used for game playing and programming in BASIC. But he had the ultimate argument: upper-case-only character generators were lots cheaper.

Though I often felt they were on the right track, I still could find myself at odds with Apple's founders, who were a strange mix of the radical and the conservative. They wanted to create personal computers, but expected them to work much like the hard-to-use minicomputers from DEC, HP, and Data General. Dragging the two Steves into the interface future was preaching in an unknown tongue, and from my perspective, they didn't appear to be the advanced thinkers that they were made out to be in the press. They were visionary, and working like mad to drag the world into the personal computer future, it's just that I was a few years further out in the future. In spite of these differences I was the typical way-over-100%-effort and totally Apple-oriented employee. This extended into my personal life. Apple's Cupertino phone number was 996-1010. When I moved to Cupertino, I chose my home phone number, symbolically, to be just one step ahead of the rest of Apple: it was 996-1009.

BITMAPPING

Apple employees were a diadem of the brightest and best cut jewels of Silicon Valley, some well known and some newly discovered. I was amazed at the competence of the people, whether in financial management, marketing, manufacturing, engineering or whatever; and they all seemed willing to share their knowledge and points of view with me. Competence clustered at Apple, partially thanks to the many contacts men like Markkula and Scott and our investors had in the industry, and partly as a result of Steve Jobs' incredible persistence. When Jobs was convinced he wanted someone, that person would be hounded to death, complimented, provided blandishments suited to his or her nature,

and offered the world. Soon enough, Apple could deliver many of these promises. At NeXT, Jobs was to continue making similar promises, repeating the ploys he had developed in Apple's first years, to the disappointment of investors, employees, and customers alike. Too often we mistake the randomness of the universe as our own accomplishment when things go our way. Still more often we take that same randomness, when it goes against us, and regard it as punishment for our sins. In a complex world it is often impossible to tell accident from design.

When Ken Rothmuller was hired from HP to start the Lisa project (which was after I had proposed the Macintosh, but before it was officially approved as a research project) I saw a new opportunity to get my computer interface and architecture ideas accepted. I argued again that the screen architecture of this new product should be bit-mapped. But where I had failed with Woz and Jobs, I managed to convince Ken and his crew -- probably to Ken's detriment as Jobs found him difficult to work with (i.e. had strong opinions and didn't kowtow) and fired him. Jobs probably found me equally difficult, but I had already proved myself and my very productive and cost-effective publications department was one of Apple's many gems; it would have been hard to justify getting rid of me.

In spite of the loss of Ken Rothmuller, the bit-mapped screen survived. This was a key win for me and (though they didn't know it at the time) for Apple, because it would force the software I was dreaming of to be implemented. No longer would computers be restricted to whatever font was in the character generator, and have to treat characters and graphics as fundamentally different kinds of things. Another major battle that I fought was to have black characters on a white background instead of the then-conventional white (or green!) lettering on a black background. The Lisa hardware designers were, like Jobs and Woz, dead set against this idea, noting that it took too much power, would require a higher refresh rate to avoid flicker, was not the way computers usually worked, and so on. I argued that people often printed computer output on white paper, and that was black-on-white, and that if you wanted it to look the same on screen and print (the WYSIWYG, or What You See Is What You Get principle) you had to do it black-on-white. But my industrial-strength argument had to do with something the Lisa crew (like the whole microcomputer industry) was just not thinking about: grayscale, or dithered, graphic images. If you worked in white-on-black and had a part-text and part-graphics image on the screen, which got reversed on printing, then either the screen or paper image would have to be a negative, and nobody wants to be forced to look at negatives.

Again, after many memos, meetings, and informal and formal discussions, I managed to sell the idea. It was another key to the future.
